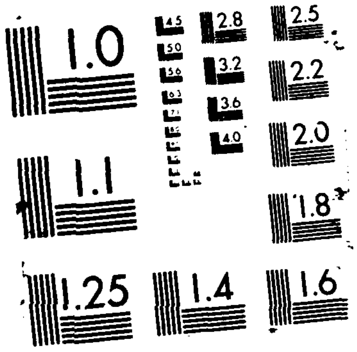
1.0
1.1
1.25
1.4
1.6
1.8
2.0
2.2
2.5
2.8
3.2
3.6
4.0

SECURITY CLASSIFICATION OF THIS PAGE

(2)

REPORT DOCUMENTATION PAGE

AD-A185 631

ILE	1b RESTRICTIVE MARKINGS DTIC FILE COPY
	3 DISTRIBUTION/AVAILABILITY OF REPORT Approved for public release; distribution unlimited.
4. PERFORMING ORGANIZATION REPORT NUMBER(S)	5. MONITORING ORGANIZATION REPORT NUMBER(S) AFOSR-TR- 87-1287
6a. NAME OF PERFORMING ORGANIZATION Inst. ~~of~~ for Scientific Computing / 6b. OFFICE SYMBOL (If applicable)	7a. NAME OF MONITORING ORGANIZATION AFOSR/NM
6c. ADDRESS (City, State, and ZIP Code) P.O. Box 1388 Fort Collins, Colorado 80522	7b ADDRESS (City, State, and ZIP Code) AFOSR/NM Bldg 410 Bolling AFB DC 20332-6448
8a. NAME OF FUNDING/SPONSORING ORGANIZATION AFOSR / 8b. OFFICE SYMBOL (If applicable) NM	9 PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER AFOSR-85-0289
8c. ADDRESS (City, State, and ZIP Code) AFOSR/NM Bldg 410 Bolling AFB DC 20332-6448	10. SOURCE OF FUNDING NUMBERS: PROGRAM ELEMENT NO. 61102F; PROJECT NO. 2304; TASK NO. A3; WORK UNIT ACCESSION NO.

11. TITLE (Include Security Classification)

Multitasked Embedded Multigrid for Three-Dimensional Flow Simulation

12. PERSONAL AUTHOR(S)
Gary M. Johnson, Julie M. Swisshelm, Daniel V. Pryor & John P. Ziebarth

13a. TYPE OF REPORT	13b. TIME COVERED	14. DATE OF REPORT (Year, Month, Day)	15. PAGE COUNT
Final	FROM ______ TO ______	June 1986	

16. SUPPLEMENTARY NOTATION

17. COSATI CODES			18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
FIELD	GROUP	SUB-GROUP	

19. ABSTRACT (Continue on reverse if necessary and identify by block number)

This project explored fast algorithms for Euler and Navier-Stokes simulations. A particular issue pursued under the grant was the integration of an explicit three dimensional flow solver, embedded mesh refinements, a model equation hierarchy, multiple grid acceleration and extensive rectorization and multi tasking. Several papers were produced during this effort, including such titles as "Multitasked embedded multigrid for three-dimensional flow simulation" and "Multigrid approsches to the Euler equations."

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT	21. ABSTRACT SECURITY CLASSIFICATION
☐ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT. ☐ DTIC USERS	[illegible]

22a. NAME OF RESPONSIBLE INDIVIDUAL	22b. TELEPHONE (Include Area Code)	22c. OFFICE SYMBOL
Maj. John Thomas	(202)767-5026	NM

DD FORM 1473, 84 MAR — 83 APR edition may be used until exhausted. All other editions are obsolete.

SECURITY CLASSIFICATION OF THIS PAGE

MULTITASKED EMBEDDED MULTIGRID FOR THREE-DIMENSIONAL FLOW SIMULATION

Gary M. Johnson, Julie M. Swisshelm,
Daniel V. Pryor and Johnny P. Ziebarth
Institute for Computational Studies
PO Box 1852
Fort Collins, Colorado 80522
U.S.A.

AFOSR-TR- 87-1287

SUMMARY

An efficient algorithm designed to be used for Navier-Stokes simulations of complex flows over complete configurations is described. The algorithm incorporates a number of elements, including an explicit three-dimensional flow solver, embedded mesh refinements, a model equation hierarchy ranging from the Euler equations through the full Navier-Stokes equations, multiple-grid convergence acceleration and extensive vectorization and multitasking for efficient execution on parallel-processing supercomputers. Results are presented for a preliminary trial of the method on a problem representative of turbomachinery applications. Based on this performance data, it is estimated that a mature implementation of the algorithm will yield overall speedups ranging as high as 100.

INTRODUCTION

It is generally recognized that a comprehensive approach to the simulation of flows involving both complex geometries and complex physics will require powerful advanced-architecture supercomputers with very large memories. Machines capable of producing solutions to Reynolds-averaged Navier-Stokes flows over complex geometries within computing times short enough to be of design interest are expected to be available by the end of this decade. In order to use these parallel-processing supercomputers effectively, algorithms must be adapted to focus the power of multiple processing units on a single flow simulation. The purpose of this work is to contribute to the development of such algorithms. The approach selected enhances the efficiency of a robust and flexible solution procedure by implementing it on a collection of local meshes embedded in a global mesh. Either the Euler, thin-layer Navier-Stokes or full Navier-Stokes equations are solved on each mesh. The choice of model equations is determined by the nature of the flow physics to be resolved on a particular mesh. When the requirement for time accuracy is relaxed, a convergence acceleration procedure is applied simultaneously to all meshes and all model equations. The entire algorithm is explicit and is designed to perform well on computers consisting of multiple processing units, each having vector processing capability. Examples of such machines are the Cray X-MP and Cray 2.

EQUATIONS OF MOTION

The nondimensional equations of motion may be written in conservation-law form as

$$q_t = -(F_x + G_y + H_z)$$

where, for the Reynolds-averaged Navier-Stokes equations,

$$F = f - \mathrm{Re}^{-1} p \qquad G = g - \mathrm{Re}^{-1} r \qquad H = h - \mathrm{Re}^{-1} s$$

while, for their thin-layer version,

$$F = f \qquad G = g \qquad H = h - \mathrm{Re}^{-1} d$$

and, for the Euler equations,

$$F = f \qquad G = g \qquad H = h$$

where:

$$q = \begin{bmatrix} \rho \\ \rho u \\ \rho v \\ \rho w \\ E \end{bmatrix} \quad f = \begin{bmatrix} \rho u \\ \rho u^2 + p \\ \rho uv \\ \rho uw \\ (E+p)u \end{bmatrix} \quad g = \begin{bmatrix} \rho v \\ \rho uv \\ \rho v^2 + p \\ \rho wv \\ (E+p)v \end{bmatrix} \quad h = \begin{bmatrix} \rho w \\ \rho uw \\ \rho vw \\ \rho w^2 + p \\ (E+p)w \end{bmatrix}$$

$$p = \begin{bmatrix} 0 \\ \tau_{xx} \\ \tau_{yx} \\ \tau_{zx} \\ \beta_x \end{bmatrix} \quad r = \begin{bmatrix} 0 \\ \tau_{xy} \\ \tau_{yy} \\ \tau_{zy} \\ \beta_y \end{bmatrix} \quad s = \begin{bmatrix} 0 \\ \tau_{xz} \\ \tau_{yz} \\ \tau_{zz} \\ \beta_z \end{bmatrix} \quad d = \begin{bmatrix} 0 \\ \mu u_z \\ \mu v_z \\ (\lambda + 2\mu) w_z \\ \gamma \kappa Pr^{-1} e_z + (\lambda + 2\mu) w w_z \end{bmatrix}$$

$$\tau_{xx} = \lambda(u_x + v_y + w_z) + 2\mu u_x \qquad \beta_x = \gamma \kappa Pr^{-1} e_x + u\tau_{xx} + v\tau_{xy} + w\tau_{xz}$$

$$\tau_{yy} = \lambda(u_x + v_y + w_z) + 2\mu v_y \qquad \beta_y = \gamma \kappa Pr^{-1} e_y + u\tau_{yx} + v\tau_{yy} + w\tau_{yz}$$

$$\tau_{zz} = \lambda(u_x + v_y + w_z) + 2\mu w_z \qquad \beta_z = \gamma \kappa Pr^{-1} e_z + u\tau_{zx} + v\tau_{zy} + w\tau_{zz}$$

$$\tau_{xy} = \tau_{yx} = \mu(u_y + v_x), \qquad \tau_{xz} = \tau_{zx} = \mu(u_z + w_x), \qquad \tau_{yz} = \tau_{zy} = \mu(v_z + w_y)$$

Here ρ, u, v, w, p and E are respectively density, velocity components in the x-, y- and z-directions, pressure and total energy per unit volume. This final quantity may be expressed as

$$E = \rho\left(e + \frac{1}{2}(u^2 + v^2 + w^2)\right)$$

where the specific internal energy, e, is related to the pressure and density by the simple law of a calorically-perfect gas

$$p = (\gamma - 1)\rho e$$

with γ denoting the ratio of specific heats. The coefficient of thermal conductivity, κ, and the viscosity coefficients, λ and μ, are assumed to be functions only of temperature. Furthermore, λ is expressed in terms of the dynamic viscosity μ by invoking Stokes' assumption of zero bulk viscosity. Re and Pr denote the Reynolds and Prandtl numbers, respectively. Although, for simplicity, the equations of motion are presented here written in Cartesian coordinates, it is well known that their strong conservation law form may be maintained under an arbitrary space- and time-dependent transformation of coordinates.

SOLUTION METHODOLOGY

The integration scheme used here is the forward predictor - backward corrector version of the two-step Lax-Wendroff method due to MacCormack [1]. This version of MacCormack's scheme is used for convenience. Any of its many variants could also be used, as could any other one- or two-step Lax-Wendroff scheme [2]. In fact, the class of fine-grid methods with which the convergence acceleration technique described below may be applied appears to be quite large, including schemes not of Lax-Wendroff type [3]. The advantages of MacCormack's method, in the present context, are its explicit nature, simplicity and low operations count. A disadvantage is its conditional stability and the severe time-step size limitation which this imposes for viscous flows, in particular. The ill effects of conditional stability are mitigated through the use of embedded grid refinements and convergence acceleration.

The embedded-mesh technique developed for the present application is a generalization of that employed in [4] to obtain two-dimensional Euler solutions. The computational domain is divided into regions requiring grids of differing fineness and the resolution of different flow physics. At present, for simplicity, this partitioning is done *a-priori*. However, solution-adaptive gridding based on this technique is possible. Fig. 1 shows typical locations for the mesh regions employed in the computations described subsequently in this paper. Note that, where mesh lines are illustrated, their spacing is much coarser than that employed in the computations. Mesh 3, the coarsest mesh in Fig. 1, covers the entire computational domain. The Euler equations are solved on it. Mesh 2, finer than mesh 3 in all directions by a factor of two, contains the regions near the walls and the blade surface where flow can be modeled by the thin-layer form of the equations of motion. The finest mesh shown, mesh 1, contains the regions of the domain near the juncture of surfaces where all viscous terms have been retained. From this specific example, it is easy to see that quite general collections of embedded meshes may be constructed in this manner. The embedded meshes are not disjoint. Rather, given a mesh labelled m, all coarser meshes from m+1 through the coarsest mesh used in the computation underlie it. This property, together with the coarsening factor of 2, facilitate the use of the multiple-grid acceleration techniques described in [5]. The flowfield updating begins with mesh 1. After one timestep on mesh 1, mesh 2 is updated exterior to mesh 1 while convergence acceleration is applied at the mesh-2 points interior to mesh 1. Next, mesh 3 is updated exterior to mesh 2 while convergence acceleration is applied at the mesh-3 points interior to mesh 2. Updating proceeds in this fashion until the global mesh has been advanced by one timestep. Then, convergence acceleration is applied to coarsenings of the global grid. This cycle is repeated until the desired measure of convergence is satisfied. Observe that when both the basic integration scheme and the coarse-grid convergence accelerator are explicit, the algorithm is particularly easy to vectorize. Additionally, a parallel coarse-grid algorithm has been developed for more efficient execution on both vector- and parallel-processing computers, as described in [5]. Further, note that, while in this paper multiple-grid convergence acceleration is applied only to steady flow simulaticns, it appears that the technique may extend to time-accurate computation of some unsteady flows [6, 7].

When implementing an algorithm on a multiple instruction-multiple data machine, we are concerned with multitasking overhead and algorithm granularity. By granularity we mean the time required to execute a multitaskable code segment on a single processor. For a given overhead, the best speedup is obtained when algorithm granularity is maximal. Large granularity is usually introduced by top-down programming which exploits global parallelism in the algorithm. Bottom-up programming exploits algorithm parallelism at a low level by making many partitionings, each on small code segments, such as DO loops containing independent statements. The sequential multigrid algorithm contains many opportunities for creating small granularity parallelism but relatively few for the sort of large granularity necessary to produce good speedup in the face of non-trivial overhead. This observation, together with the desirability of non-sequential multigrid schemes for reasons of algorithm flexibility, led to the construction of the parallel multigrid algorithms mentioned above. In these algorithms, grids which are independent of one another may be updated simultaneously on separate processors. In fact, such a simple strategy may result in a poor load balance across processors because of the differing amounts of work inherent in updating grids of different coarseness. However, more refined strategies are possible. Grids may be grouped together into tasks of approximately equal work, or they may be melded into tasks with other large-grained code segments in order to equalize processor loading. Notice futher that, by multitasking large-grained structures, the vectorization potential of code within these structures remains intact.

NUMERICAL SIMULATION

As the algorithm described in this paper is designed to efficiently simulate complex flows over complete configurations, it should be tested under conditions which fully exercise its capabilities. On the other hand, excessive complexity would serve no useful purpose in the initial testing phases of the algorithm. With these considerations in mind, three-dimensional computations are being carried out for the geometry illustrated in Fig. 2, a rectilinear cascade of finite-span, swept blades mounted between endwalls. The sweep angle ranges from 0 to 26 degrees. The blade thickness to chord ratio ranges from 0.0 to 0.2. The subcritical computations are performed at an isentropic inlet Mach number of 0.5. The Mach number for the supercritical computations is 0.675. In the viscous cases, the Reynolds numbers, based on cascade gap and critical speed, span the approximate range from 8.4×10^3 to 2.0×10^5. The mesh structure on which the computations are being performed is illustrated in Fig. 1. The full Navier-Stokes equations are solved on mesh 1. The thin-layer Navier-Stokes equations are solved on mesh 2. The Euler equations are solved on mesh 3. Only steady flows are computed and convergence acceleration, as described previously, is applied. The entire algorithm is vectorized and multitasked to run on a four-processor Cray X-MP or Cray 2. Sample results for a subcritical flow over a swept blade are shown in Fig. 3.

Comparison of the embedded-mesh algorithm with a single-mesh algorithm yields the following conclusion: the accuracy of the embedded-mesh results is essentially that of a global finest mesh, while the convergence rate is like that of a global coarsest mesh. Thus far, in two-dimensional computations using the Euler and thin-layer Navier-Stokes equations and three mesh regions, embedding speedups as high as 30 have been obtained. Three-dimensional embeddings using Euler, thin-layer and full Navier-Stokes regions should produce substantially larger speedups.

Multiple-grid convergence acceleration applied to three-dimensional cases, in the absence of mesh embedding, has yielded speedups ranging from 2.5 to 4.7. It is expected that there will be some tradeoff between embedding and multigrid speedup in the complete algorithm. Vectorization of the three-dimensional algorithm without embedded meshes results in speedups ranging from 3.6 to 5.7. This range should remain about the same in the final algorithm.

Using a top-down multitasking approach, the parallel coarse-grid algorithm has been implemented on a four processor Cray X-MP, for two-dimensional cases without mesh embedding. Initially, only the coarse grids were multitasked so that the performance of parallel grids on a multiprocessor could be evaluated. Then the fine-grid computations were partitioned and multitasked, and the resultant code was integrated with the parallelized coarse grids.

For the multitasking results, performance measures are based on a comparison of multitasked code segments with their unitasked analogs. The parallel coarse-grid scheme results were obtained with a five-grid multigrid sequence length. An efficiency of nearly 90% has been obtained using two processors, but that efficiency deteriorates to 77% when four processors are used. This deterioration is a result of distributing multigrid structures containing unequal amounts of work across four processors. Results obtained from multitasking the fine-grid scheme show that the fine-grid tasks are fairly evenly balanced, and this code segment performs well on both two and four processors. Processor utilization of 90% or better is achieved. The fully multitasked two-dimensional multigrid algorithm attains efficiency levels ranging from 94% on two processors to 83% on four processors. For the three-dimensional Navier-Stokes code, the bottom-up approach is taken by using microtasking software on the Cray X-MP. Microtasking incurs relatively low overhead, which allows parallelization of very fine-grained code segments and alleviates the need for careful *a-priori* load balancing. The resulting fully microtasked three-dimensional code performance ranged from 98% efficiency on two processors to 89% on four processors.

Given that the speedups from the various categories described above are generally multiplicative in effect, it is to be conservatively estimated that a mature implementation of the algorithm will produce overall speedups ranging as high as 100.

CONCLUSIONS

An efficient algorithm designed to be used for Navier-Stokes simulations of complex flows over complete configurations has been presented.

The algorithm makes use of several elements: a robust explicit basic flow solver, locally-embedded mesh refinements, a flow *simulation* hierarchy ranging from the Euler equations through the full Navier-Stokes equations, an explicit multiple-grid convergence acceleration technique, and both vectorization and multitasking for efficient execution on parallel-processing supercomputers.

Results are presented for a problem representative of turbomachinery applications. These results provide grounds for optimism regarding the algorithm's future application to more challenging internal and external flows. Based on the performance data presently available, this algorithm is expected to reduce simulation times by as much as two orders of magnitude.

ACKNOWLEDGEMENTS

This research is funded in part by the NASA Ames Research Center (NCC-2-344) and the Air Force Office of Scientific Research (AFOSR-85-289). Some of the computing time has been contributed by Cray Research, Inc. All of this support is gratefully acknowledged.

REFERENCES

1. MacCormack, R.W.: AIAA Paper 69-354, 1969.
2. Johnson, G.M.: NASA TM-82843, 1982.
3. Stubbs, R.M.: AIAA Paper 83-1945, 1983.
4. Usab, W.J.: Doctoral Dissertation, Aero and Astro Dept., MIT, 1983.
5. Johnson, G.M., Swisshelm, J.M. and Kumar, S.P.: AIAA Paper 85-1508, 1985.
6. Stubbs, R.M.: Private Communication, 1983.
7. Jespersen, D.C.: AIAA Paper 85-1493, 1985.

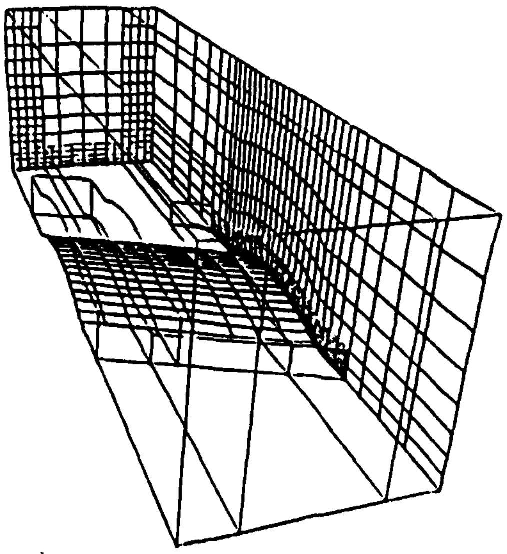

Figure 1. Typical Locations for Embedded Mesh Regions

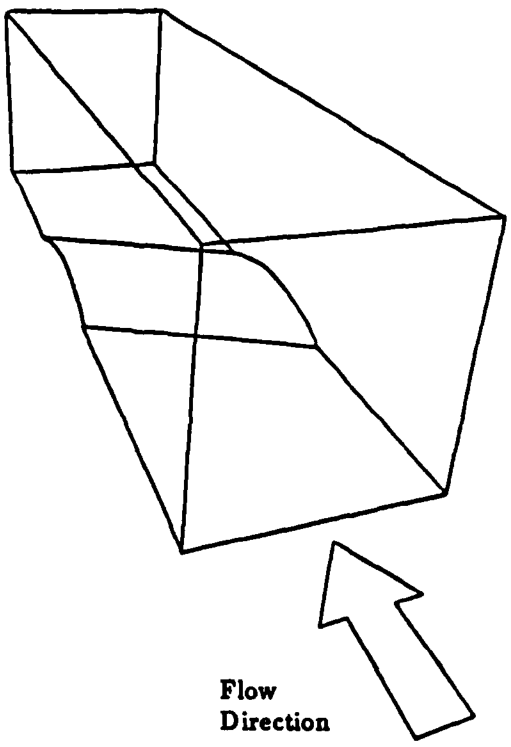

Figure 2. Computational Domain

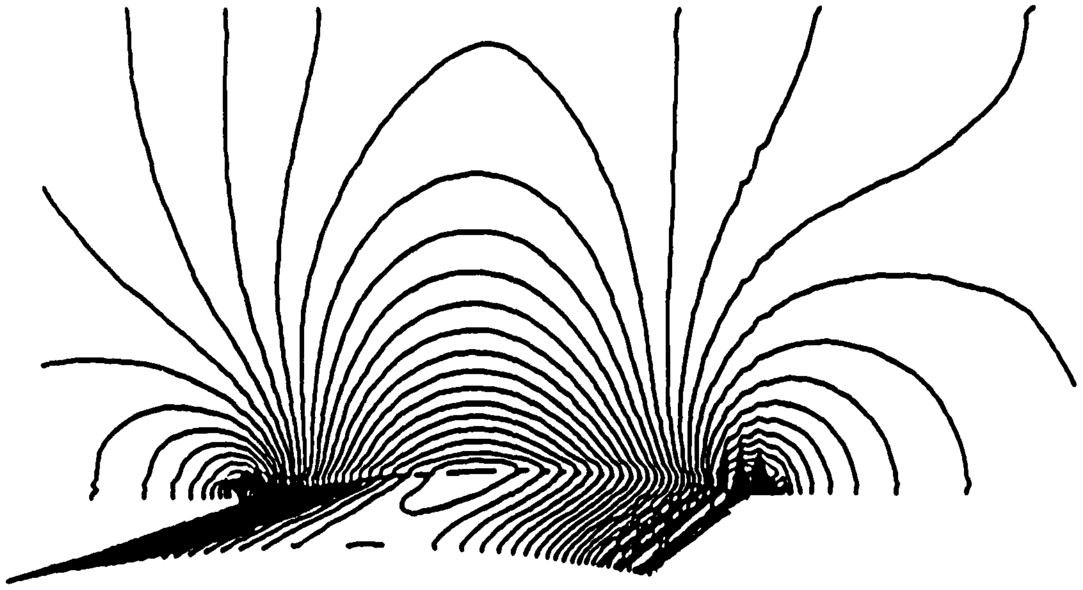

Figure 3. Isobars for Subcritical Flow over Swept Blade